International Journal of Engineering & Technology, 7 (4.41) (2018) 86-90

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

An Efficient Approach to Detect Driver Distraction during Mobile Phone Usage

Medha Sawhney[1], Vasundhara Acharya[2*], Krishna Prakasha[3]

[1]Department of ECE, Manipal Institute of Technology , Manipal Academy of Higher Education, Manipal , INDIA
[2#] Department of CSE, Manipal Institute of Technology , Manipal Academy of Higher Education, Manipal , INDIA
[3]Department of I&CT,, Manipal Institute of Technology , Manipal Academy of Higher Education, Manipal , INDIA
**Corresponding author E-mail:vasundhara.acharya@manipal.edu*

Abstract

A distracted driver is an invitation to a fatal vehicle accident. People lose their lives every day due to distracted driving and using mobile phones while driving is one of the primary reasons behind road accidents. Hence, detection of mobile phone usage to alert the driver or for an autonomous system to take over becomes extremely important. In an attempt to solve this issue of distracted driving, the authors proposed a Convolution Neural Network (CNN) based model to detect mobile phone usage by the driver. The proposed work presents not only a practical solution to the problem but also a comparison between traditional approaches (Support Vector Machine with HOG) and a CNN based model. The traditional methods are both implemented and tested by the authors. The presented model performs input segmentation to achieve an efficient accuracy of 97%. Deep learning was found to be the best solution to detect driver distraction while on a call accurately

***Keywords**: Convolutional Neural Network; Distracted Driver detection; Mobile phone usage of driver; Image classification; Object detection.*

1. Introduction

Millions of people are losing their lives every day due to road accidents [1][2]. The primary cause of road accidents is distracted driving. Distracted driving happens when the driver is engaged in an activity that is not driving, and that needs them to focus on that making them inattentive to the driving task. According to a 2014 study conducted by the National Highway Traffic Safety Administration (NHTSA), 3,179 people were killed, and 431,000 drivers were injured in motor vehicles crashes due to distracted drivers. [3].Multiple distractions can be present for a driver like drunk driving fatigue, depression or anxiety. One of the most dangerous yet addictive distraction while driving is the use of cell phones [4] for talking or texting. According to a report conducted by the NHTSA in 2013, talking on phone, texting, operating the radio or grooming activities requires both cognitive and visual attention, and hence it increases the chance of an accident by nearly a factor of 3 [5]. This work focusses on detecting distraction due to handheld cell phone usage during driving vehicles. Driver distraction detection can be done using necessary image processing on the frontal image on the driver while driving. Advanced image classification and machine learning techniques like Convolutional Neural Networks (CNN), Recurrent or Recursive Neural Networks can be used to detect driver distraction. The fundamental approach of image processing such as Support Vector Machines(SVM) can also be implemented to solve this problem. The proposed work presents a simple solution to detect distracted driving using an in-car camera to get the frontal image of the driver. This algorithm could be used with any simple primary alarm system to alert the driver that he/she has not been focusing on the road. It can also be used in autonomous cars to shift control to the machine in case the driver is found inattentive. The authors have implemented a basic CNN model to detect mobile phone distraction with accuracy score above 95%. Insight into preparing the appropriate data set for the CNN model to give maximum accuracy is provided in the proposed work. The output of the CNN model is a binary class label and the probability of the input image. Comparative analysis based on experimental data, between CNN and SVM with HOG feature extraction is also presented. Related work on this problem statement has been discussed in detail in Section 2. Section 3 describes the complete methodology of the proposed work. Comparative analysis of all the experimental results and the related techniques have been discussed in Section 4 .

2. Related Works

Colbra et al.. [6] successfully implemented combinations of CNN based models and achieved accuracies between 80% to 86 %.
In [7] the authors used a pre-trained VGG-16 model. Caffe's suggestion was used for the learning rate. They have also implemented a GoogleNet to solve the same problem.
Streiffer et al. [8] succeeded in detecting multiple distractions like texting and hair or makeup using various combinations of CNN with RNN or SVM but maximum accuracy obtained was 87%. In addition to using deep learning, they made use of embedded mobile sensors. Both Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) were implemented. Classification on a per-frame basis was done using CNN, and the time-series data was classified using RNN.
Hssayeni [9] presented a comparison between old techniques of image classification specially SVM (Support Vector Machines) with HOG (Histogram of Gradients) feature and CNN models. The best accuracy achieved in implementing CNN was 85%.

R. Berri et al. [10] used a pattern recognition system (PR) for classification using Artificial Neural Networks (ANN). They made use skin segmentation technique in their model.
H. Yasar [11] implemented a cascade classifier to detect mobile phone distraction. Object detection of the driver's hands and fingers near his/her head was proposed.
K. Seshadri [12] detected facial landmarks to identify the face and to extract the ROI. They implemented feature extraction on the ROI and used the Real AdaBoost framework for classification. R. A. Berri et al. [13] implemented an SVM based model using the extraction of skin pixels to detect the percentage of hand present in the ROI. They also calculated the MI (moment of inertia) to differentiate between the image of holding a cell phone and of normal driving.

3. Methodology

The system framework is described in this section. Preparation of data is a highly crucial step in all machine learning problems. Dataset preparation and the training method is discussed here. The proposed work follows CNN approach. The authors have also experimented on SVM with HOG and Random Forests for the same problem, contributing to a comparative analysis mentioned in this paper. The steps followed in the system are given in brief in algorithm1. Fig.2. depicts the flowchart of the proposed model.

Algorithm 1:
Input: Frontal image of the driver's face, driving a vehicle, clicked by an in-car camera.
Output: Class 1 or 0. 0 for Normal Driving and 1 for Phone Distracted Driving
Step 1: Perform face detection on the input image to get face rectangle coordinates. The image is not processed further if face is not detected.
Step 2: To generate ROI, plot two rectangles (right and left of face rectangle) on the input image using the face rectangle coordinates. The result is a region that covers ear and portion around the ear. ROI rectangles are plotted to be of the same size but size can vary depending on where the face is present in the image.
Step 3: For ROI extraction crop both side rectangles and store it to use as input to CNN model. All images obtained after ROI extraction are brought up to the same size.
Step 4: Execute CNN model with input as ROI images of two ears. The output generated is class labels 1 or 0 or probabilities of the image classification.
END

3.1. Data set preparation

The data set is prepared using an in-car camera mounted such that it is facing the driver while driving. Frames are extracted from the videos obtained from the camera during different driving incidents in the vehicle. The frames captured are read in grayscale to enhance feature extraction. The dataset prepared for this paper has 10,000 training images, 8,000 test images, and 4,000 validation images.

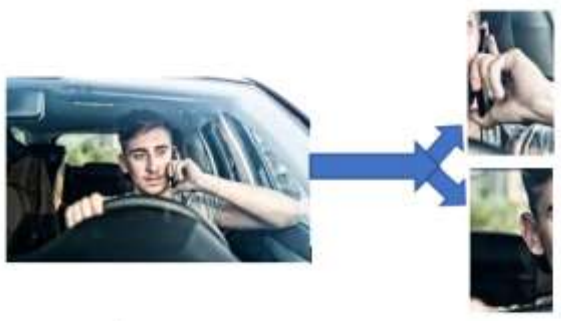


Fig. 1: ROI extraction of frontal image of driver

3.2. ROI segmentation

Face detection is performed on the input images, and a rectangle bounding the face is obtained. Using the coordinates of this face rectangle, two rectangles are plotted on the input image, one on each side of the face near the ear, as shown in Fig. 1. Then these rectangles are cropped and stored. The obtained ROI (region of interest) is then used as input to the CNN model.

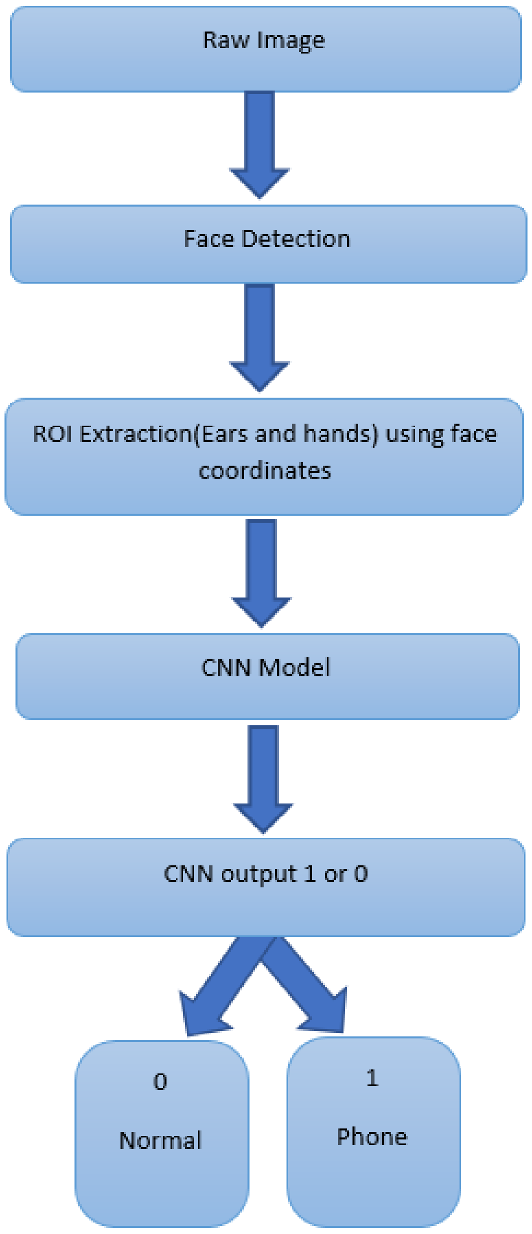


Fig. 2: Flowchart of the proposed model

3.3. Convolutional Neural Networks (CNN)

Convolution refers to the mathematical combination of two functions to produce a third function [14]. It merges two sets of information. In the case of a CNN, the convolution is performed on the input data with the use of a filter or kernel to produce a feature map. Convolutional neural networks comprise one or more layers of neurons with weights and biases attached to each neuron. The initial values of the weights and biases are assumed in the beginning and are later learned by the neural network gradually. Each network has a input layer, an output layer and hidden layers in between that do the learning. Each neuron takes an input and performs a dot product which is usually ensured by the application of a non-linear function. The network as a whole expresses a single differentiable score function having raw image pixels at the input end and class scores at the output end [15]. Convolutional neural networks also implement a loss function at the end of the last fully connected layer. Convolutional networks make an explicit assumption that the inputs are images, which gives the allowance to encode specific properties peculiar to images in a CNN model. These properties make the forward function highly efficient and also reduce the number of parameters in the network. Fully connected layers connect every neuron in one layer to every neuron in another layer. The last fully-connected layer uses a SoftMax activation function for classifying the generated features of the input image into various classes based on the training dataset. Fig.3. Shows the block diagram of one of the CNN models experimented and tested. Fig.4. Depicts the summary of the CNN model.

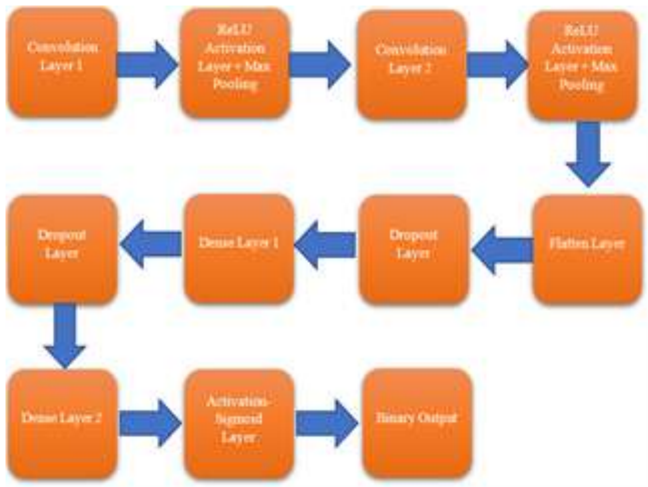


Fig. 3: Block Diagram of the layers used in the CNN model with 2 layers

3.3.1 Convolution Layer

The convolution layer is the basic fundamental block of a CNN. It comprises a set of filters that independently convolve with the input image and output feature maps. Feature Map is the output volume formed by sliding the filter over the image and computing the dot product. These filters are initialized randomly and are learned by the CNN gradually [16], [17].

3.3.2 Max Pooling Layer

A pooling layer is used to reduce the spatial dimensions of the input and also the computational complexity of the CNN model. When the images or in between layer matrices are too large, pooling layer is used to reduce the number of trainable parameters. It also controls overfitting. This layer operates independently on every depth slice of the input. Different types of pooling functions exist such as Max pooling, average pooling, or L2-norm pooling. However, Max pooling is the most commonly used type of pooling. It is a sample-based discretization process to down-sample an input image or matrix, reducing its dimensionality and allowing for assumptions to be made about features contained in the sub-regions binned. It is then desired to introduce pooling layers between sub-sequent convolution layers periodically. Pooling is done for the sole purpose of reducing the spatial size of the image. Pooling is done independently on each depth dimension. Therefore the depth of the image remains unchanged. It is also referred to as a down-sampling layer.

3.3.3 Dropout Layer

Dropout Layer has a crucial function in neural networks. This layer "drops out" a random set of activations in the previous layer by setting them to zero. This layer ensures that overfitting on the training data is avoided or reduced. Dropping out nodes is useful because it prevents inter-dependencies emerging between nodes. It allows the network to learn a more robust relationship. Implementing dropout has much the same effect as taking the average from a committee of networks. However, the cost is significantly less in both time and storage required.

3.3.4 Dense Layer

Dense Layer performs classification on the features extracted by the convolutional layers and down-sampled by the pooling layers. It is used to change the dimensions of your vector. Dense layer applies a rotation, scaling, translation transform to your vector. It is termed a Fully Connected Layer because, in a dense layer, every node in the layer is connected to every node in the preceding layer [18].

3.3.5 Flatten Layer

Flatten layer is used to convert all the resultant two-dimensional arrays into a single continuous linear vector. The flattening step is needed to make use of fully connected layers after some convolutional layers. Fully connected layers don't have a local limitation like convolutional layers (which only observe some local part of an image by using convolutional filters). This layer combines all the local features found by the previous convolutional layers. Each feature map channel in the output of a CNN layer is a "flattened" two-dimensional array created by adding the results of multiple two-dimensional kernels (one for each channel in the input layer).

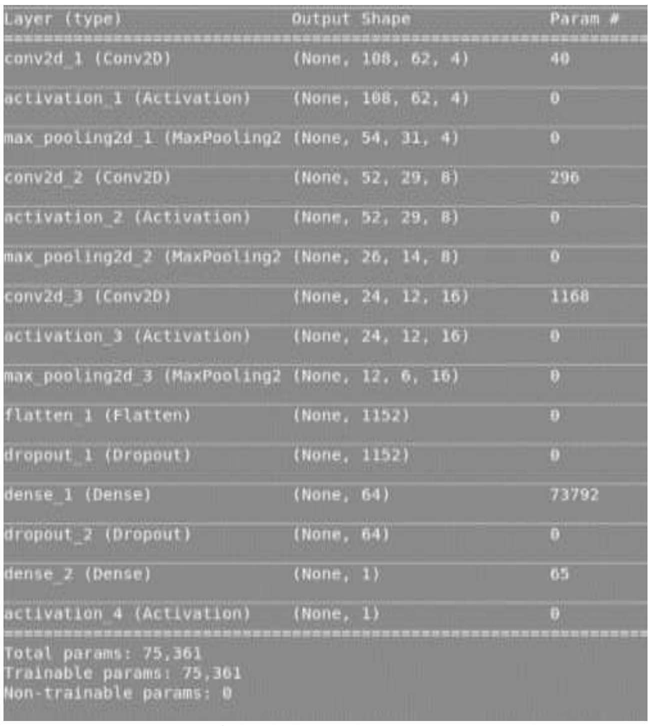
3.3.6 Sigmoid activation function

Sigmoid is an activation function as shown in (1). It ranges between 0 and 1. It produces a curve with an "S" shape. In general, a sigmoid function is a real-valued and differentiable function having a non-negative or non-positive first derivative with one local minimum and only one local maximum. A Sigmoid function is used in artificial neural networks to introduce nonlinearity in the model. It is applied to the result of the neural network model. The output of the sigmoid function when x=0 is 0.5. Thus, if the output is more than 0.5, we can classify the outcome as 1, and if it is less than 0.5, we can classify it as 0.

$$s(t) = \frac{1}{1 + e^{-t}} \qquad (1)$$

3.3.7 ReLU activation function

ReLU Layer implements the elementwise rectifier activation function f(x) = Max(0, x). The function threshold is at zero, converting all negative values to zero. ReLU can be efficiently computed compared with other activation functions like the sigmoid and hyperbolic tangent, without making a significant difference to generalization accuracy. It is used in place of a linear activation function to add non-linearity to the neural network. ReLU layer does not change the size of the volume. It also helps to alleviate the vanishing gradient problem (which occurs in sigmoid activation function), which is the issue where the lower layers of the network train very slowly because the gradient decreases exponentially through the layers.



```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 108, 62, 4)        40
_________________________________________________________________
activation_1 (Activation)    (None, 108, 62, 4)        0
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 54, 31, 4)         0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 52, 29, 8)         296
_________________________________________________________________
activation_2 (Activation)    (None, 52, 29, 8)         0
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 26, 14, 8)         0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 24, 12, 16)        1168
_________________________________________________________________
activation_3 (Activation)    (None, 24, 12, 16)        0
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 12, 6, 16)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 1152)              0
_________________________________________________________________
dropout_1 (Dropout)          (None, 1152)              0
_________________________________________________________________
dense_1 (Dense)              (None, 64)                73792
_________________________________________________________________
dropout_2 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 65
_________________________________________________________________
activation_4 (Activation)    (None, 1)                 0
=================================================================
Total params: 75,361
Trainable params: 75,361
Non-trainable params: 0
```

Fig. 4: Summary of the CNN model used in the proposed work

3.5 Support Vector Machines (SVM)

SVM is a supervised machine learning algorithm. It is a discriminative classifier which is defined by a separating hyperplane. It can be used for classification or regression problems. Classification is performed by finding the hyperplane that maximizes the margin between the two classes in case of a two-dimensional input plane. The vectors that define the hyperplane are called support vectors; they are the extreme points in the input dataset. The input of an SVM is labeled train data, and the algorithm outputs a hyperplane that divides the input into two categories. In a two dimensional plane, the hyperplane is just a line separating the input plane into two parts.

3.6 Histogram of Gradients (HOG)

HOG is a dense feature extraction method for images. Dense implies that it extracts features for all locations in the image. It is widely used as an efficient feature described image for object detection. The output of HOG is a feature descriptor. A feature descriptor is a representation of an image or an image patch that simplifies the image by extracting useful information and throwing away extraneous information. HOG is generally used as input to some other machine learning model like SVM. HOG tries to capture the shape of structures in the region by obtaining information about gradients. It does so by dividing the image into small (usually 8x8 pixels) cells and blocks of 4x4 cells. Each cell has a fixed number of gradient orientation bins. Each pixel in the cell votes for a gradient orientation bin with a vote proportional to the gradient magnitude at that pixel. Gamma and colors of the image should be normalized to increase the accuracy of the feature image obtained which improves the efficiency of object detection [19].
The authors have tested an implementation of a SVM+HOG model for mobile phone distraction detection with an accuracy of 90% on test data. HOG was implanted on the ROI generated and not on the raw input image.

3.7 Random Forests

Random Forest is a collection of multiple decision trees. Random Forest algorithm is used for both classification and regression problems. It creates a forest with some trees. In general, with the increase in the number of trees, the forest becomes more robust. In the same way in a random forest classifier, the higher the number of trees in the forest gives the high accuracy results. Two essential terms in Random Forest implementation are Randomness and Diversity. Each decision tree in every row of Random Forest gets a random sample of the training data. So each of these decision trees is getting trained on a randomly chosen part of the dataset, and so they will be different from each other regarding predictions. After each row is processed, not all the outputs from all decision trees are passed into training each of the decision trees that follow.
Random Forests do not give good results on a smaller dataset; they need a considerable dataset to provide efficient results. Random Forests also take a more significant time to train comparatively.
The authors tested a Random Forest implementation for the current problem statement, and they were able to successfully detect driver distraction with an accuracy of 92.4% on test data.

4. Results

This section discusses all the experiments performed by the authors for the problem at hand and also compares and discusses related works and how they could be improved. The equation used for accuracy is given in (2), where TP is True Positive, TN is True Negative, FP is False Positive, and FN is False Negative. A true positive is when a model correctly predicts the positive class, that is when a driver is distracted and is detected as distracted. Similarly, a true negative is when the model accurately predicts the negative class, focused driving is identified as normal driving in the proposed model. While, a false positive is when the model incorrectly predicts the positive class, a distracted driver is declared attentive. A false negative is when the model incorrectly predicts the negative class when a focused driver is reported to be distracted [20].

$$\text{Accuracy} = (TP + TN) / (TP + FP + TN + FN) \tag{2}$$

4.1 Experiments

The proposed work presents a CNN based model. The proposed work has also implemented other approaches to solve the problem statement and achieve comparative analysis. The authors have applied and tested an SVM model using HOG features and even a model based on Random Forests. Linear SVM works well for classification problems, and the efficiency of the SVM based model can be increased by giving a feature vector as an input instead of the raw image. The linear SVM +HOG model gave an accuracy of 90% on test data and 85% on validation data. The Random Forest implementation gave an efficiency of 92.4% on test data. CNN model had the best performance on the collected data. CNN model with three layers had slightly more accuracy than the model with two layers. The proposed model had a True positive 0f 3982 and a True Negative of 3770 out of 4000 test images for both positive and negative. CNN model with two layers performed decently with an accuracy of 93% on test data while the model with three layers had an accuracy of 96.9% on test data. The confusion matrix for the proposed and tested CNN model is given in Table 1, and the comparative analysis of the experiments performed is tabulated in Table 2.

Table 1: Confusion matrix for the presented CNN model

Actual/Predicted	Positive	Negative	Accuracy
Positive (4000)	3982	18	99.5%
Negative (4000)	230	3770	94.25

Table 2: Comparative analysis

Approach	Test Data Accuracy (%)	Validation Data Accuracy (%)
SVM (linear) +HOG	90	85
Random Forest	92.4	84
CNN (3 Convolutional	96.9	89.7

Layers)		
CNN (2 Convolutional Layers)	93.4	82

4.2 Comparative Analysis

A comparative analysis of the state of the art techniques is discussed here. RNN based models have been used commonly in the related works. RNN helps to solve a lot of machine learning problems. CNN works better for image classification while RNN performs better on the time data. Segmentation of the input image is also a crucial step in object detection and classification. Using only the ROI and not the raw image contributes to the efficiency of the algorithm.

Table 3: Comparative analysis of all related works

Approach	Accuracy (%)	Drawbacks	References
CNN combinations VGG-16	80-86	Image segmentation or feature extraction could be tried to improve accuracy	[6]
CNN+RNN	87.02%		[8]
CNN	85%	Image segmentation should be tried to improve accuracy.	[9]
ANN+skin segmentation with PR	91.68%	--------------------	[10]
Cascade Classifier	75%	Detection depends on vehicle type	[11]
Real Adaboost	93.86%	-------------------	[12]
SVM with MI and PH	91.57%	---------------------	[13]

5. Conclusion

The proposed work presented an algorithm which allows detection of the use of cell phones by drivers while driving a vehicle using the frontal image of the driver. Not a lot of research has been done on this subject as mostly the focus is on drowsiness detection, posture detection, gaze detection or mood detection. The proposed model implements segmentation of the input image of the driver and uses that as input to a CNN based model. Segmentation of ROI with useful classification results has been performed by the authors. The output of the CNN model is the probabilities of image classification and class labels that have a best-case accuracy of 96.9%. Ideas to enhance or improve this research work have been discussed in the next section.

6. Future Scope

The authors want to detect other forms of mobile distraction also such as texting, or attending a phone call using earphones. The proposed work is limited to face detection. The authors would want to improvise it such that if an extreme angle of the face is obtained and one of the ears is visible, mobile distraction detection can be still done.

References

[1] M. A. Regan, J. D. Lee, and K. L. Young, Driver distraction: Theory, effects, and mitigation. Boca Raton, FL, USA: CRC Press, 2008.

[2] M. Peissner, V. Doebler, and F. Metze, “Can voice interaction help reducing the level of distraction and prevent accidents?” 2011

[3] 2016. Research Note: Distracted Driving 2014. (Apr 2016). https://crashstats.nhtsa.dot.gov/api/public/viewpublication/812260

[4] D. L. Strayer, F. A. Drews, and D. J. Crouch. A Comparison of the Cell Phone Driver and the Drunk Driver. Human factors: The Journal of the Human Factors and Ergonomics Society, 48(2):381–391, 2006

[5] Gregory M Fitch, Susan A Soccolich, Feng Guo, Julie McClafferty, Youjia Fang, and others. 2013. The impact of hand-held and hands-free cell phone use on driving performance and safety-critical event risk. Technical Report.

[6] Colbran, Samuel, Kaiqi Cen, and Danni Luo. "Classification of Driver Distraction."

[7] BVLC. Solver : http://caffe.berkeleyvision.org/tutorial/solver.html

[8] Streiffer, C.; Raghavendra, R.; Benson, T.; Srivatsa, M. DarNet: A Deep Learning Solution for Distracted Driving Detection. In Proceedings of the Middleware Industry 2017, Industrial Track of the 18th International Middleware Conference, Las. Vegas, NV, USA, 11–15 December 2017.

[9] Hssayeni, Murtadha D; Saxena, Sagar; Ptucha, Raymond; Savakis, Andreas, “Distracted Driver Detection: Deep Learning vs Handcrafted Features”, Society for Imaging Science and Technology, Imaging and Multimedia Analytics in a Web and Mobile World 2017, pp. 20-26(7)

[10] R. Berri, F. Osório, R. Parpinelli and A. Silva, "A hybrid vision system for detecting use of mobile phones while driving," *2016 International Joint Conference on Neural Networks (IJCNN)*, Vancouver, BC, 2016, pp. 4601-4610.

[11] H. Yasar, "Detection of Driver's mobile phone usage," *2017IEEE 9th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM)*, Manila, 2017, pp. 1-4.

[12] K. Seshadri, F. Juefei-Xu, D. K. Pal, and M. Savvides. Driver Cell Phone Usage Detection on Strategic Highway Research Program (SHRP2) Face View Videos. In Computer Vision and Pattern Recognition Workshops (CVPRW), 2015 IEEE Conference on, June 2015.

[13] R. A. Berri, A. G. Silva, R. S. Parpinelli, E. Girardi, and R. Arthur, “A pattern recognition system for detecting use of mobile phones while driving,” arXiv preprint arXiv:1408.0680, 2014.

[14] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016, http://www.deeplearningbook.org.

[15] Supervised Convolution Neural Network : http://ufldl.stanford.edu/tutorial/supervised/ConvolutionalNeuralNetwork/. Accessed on : 2018-08-27.

[16] Convolution Neural Networks: http://cs231n.github.io/convolutional-networks/#overview Accessed on : 2018-08-27.

[17] Classification based on Deep Convolutional Neural Networks: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf Accessed on : 2018-08-27.

[18] Core Layers on CNN in Keras: https://keras.io/layers/core/ Accessed on : 2018-08-27.

[19] Dalal, Navneet, and Bill Triggs. "Histograms of oriented gradients for human detection." Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. Vol. 1. IEEE, 2005.

[20] True Positive and True Negative: https://developers.google.com/machine-learning/crash-course/classification/true-false-positive-negative Accessed on : 2018-08-27.